PCT/ SE 99 / 0 1 9 2 3

4

PRV
PATENT- OCH REGISTRERINGSVERKET
Patentavdelningen

SE99/1923

Intyg
Certificate

Härmed intygas att bifogade kopior överensstämmer med de handlingar som ursprungligen ingivits till Patent- och registreringsverket i nedannämnda ansökan.

This is to certify that the annexed is a true copy of the documents as originally filed with the Patent- and Registration Office in connection with the following patent application.

(71) Sökande Imsys AB, Järfälla SE
Applicant (s)

(21) Patentansökningsnummer 9803708-8
Patent application number

(86) Ingivningsdatum 1998-10-28
Date of filing

Stockholm, 2000-01-03

För Patent- och registreringsverket
For the Patent- and Registration Office

Emma Johnsson
Emma Johnsson

Avgift
Fee

CONTROLLING ACCESS TO A DYNAMIC RANDOM ACCESS MEMORY

TECHNICAL FIELD OF THE INVENTION

The present invention relates to a method for controlling access to a dynamic random access memory (DRAM), as well as to a DRAM controller and a computer system incorporating a DRAM controller.

BACKGROUND OF THE INVENTION

In most digital processing systems, such as microprocessor-based computer systems, the main memory is a DRAM. The term "DRAM" is an acronym for "Dynamic Random Access Memory." "Dynamic" indicates that for the memory to remember data, the memory requires every bit to be refreshed within a certain time period. As is well known, data is stored capacitively in a DRAM and hence data must be regenerated regularly to avoid data losses due to capacitive leakage. When the power is removed from the DRAM, the data is lost. "Random Access" indicates that each cell in the memory can be read or written in any order. This contrasts a sequential memory device where data must be read or written in a certain order. The reason behind the popularity of DRAMs for high-capacity memories in digital processing systems is the low cost per bit due to the high memory density.

The memory cells of a DRAM are arranged in rows and columns, and a given memory cell in the DRAM is accessed by first applying a row address and then applying a column address to the DRAM. The row address is strobed into the DRAM by activating a row address strobe (RAS) signal, thus energizing the selected row. The column address is strobed into the DRAM by activating a column address strobe (CAS) signal, and data is selected from the specified column in the energized row. In general, the DRAM address is multiplexed into the DRAM such that the same terminals are used, at different times, for the row address and the column address.

Access to the DRAM is normally controlled by a DRAM controller that generates the necessary control and address signals to the DRAM and determines the sequence and relative timing of these signals.

In general, processor cores are designed to suit simple memories, such as static random access memories (SRAMs), sometimes used as cache memories. Traditionally, the processor-memory interface is designed in such a way that, for each memory access, the processor has to decide whether a read or write operation is to be executed, and also has to determine a complete memory address to be used before initiating the read or write access to the memory. For a DRAM, which is a rather complex memory with a multiplexed addressing procedure, this generally means that the DRAM will be relatively slow with long access times. In this respect, DRAMs are often considered as the bottlenecks of digital processing systems.

A common way of alleviating this problem is to use a small but fast cache memory as a buffer between the processor and the larger and slower DRAM. The cache memory includes copies of small parts of data and/or program information stored in the DRAM. When the processor has to read a memory cell not available in the cache memory, that cell will be copied together with a number of adjacent cells from the DRAM to the cache memory. For subsequent accesses to these cells, the processor then interfaces the fast cache memory instead of the DRAM.

RELATED ART

Attempts have been made in the prior art to reduce the access times for DRAMs:

The international patent application WO 96/37830 relates to a microprocessor with a pipelined access request to an external DRAM. Memory requests are pipelined to the external DRAM by calculating a memory address during the same clock cycle that the instruction associated with the address is latched by an execution stage, issuing an early ready signal and directing the information received from the external DRAM to a register file during the same clock cycle that the information is received.

The international patent application WO 96/29652 relates to a rule-based DRAM controller. A DRAM controller asserts memory access and control signals based on predefined control rules. Certain rules are used to determine the timing and sequence of the memory accessing signals and control signals output by the DRAM controller. The control rules are implemented as logic within the controller, while the conditions for the rules are provided from various monitoring signals and timing modules. Based upon these rules and conditions, the request inputs to the controller are interpreted to provide optimum access speed to the DRAM.

The international patent application WO 96/30838 relates to a DRAM controller. The DRAM controller is designed to reduce the access time for read and write cycle accesses in a memory system with page mode accesses, by rearranging the processing order of the read and write cycle access requests.

SUMMARY OF THE INVENTION

It is a general object of the present invention to reduce the time required for accessing a DRAM.

It is an object of the invention to provide a method for controlling access to a DRAM such that access times comparable to those of SRAMs are obtained.

Yet another object of the invention is to provide a DRAM controller, preferably incorporated in a microprocessor-based computer system, which provides fast access to a DRAM.

These and other objects are met by the invention as defined by the accompanying patent claims.

In accordance with the invention, the access control for the DRAM is intimately associated with the micro code instructions of the processor. The interface between the processor and the DRAM controller is customized in such a way that the memory access control is integrated into the micro code program of the processor. Each micro code instruction of the processor includes a control instruction that is used in controlling the operation of the DRAM controller. More particularly, the DRAM controller controls access to the DRAM by executing, for each DRAM access, a sequence of DRAM control operations in response to a corresponding sequence of control instructions included in the micro code instructions of the processor.

It is important to understand that the control instruction included in each micro code instruction only constitutes a part of the micro code instruction, and that the remaining part of the micro code instruction is used by the processor for the control of other operations such as determining address information for the DRAM and processing data to/from the DRAM.

Since each DRAM access is based on a sequence of DRAM control operations, each of which is controlled by a respective control instruction in a micro code instruction, a DRAM access can be initiated by performing the first DRAM control operation in the sequence, without having all the information required for subsequent DRAM control operations in the sequence. The information required for a subsequent DRAM control operation is preferably determined by the current micro code instruction in parallel with the execution of the first DRAM control operation.

For example, the micro code program can initiate a DRAM access before it has been decided whether a read or write access should be performed, and the row address can be applied to the DRAM before the column address has been determined. This substantially reduces the access time for a DRAM access.

Each control instruction determines which one of a number of predefined DRAM control operations that the DRAM controller will execute. Under the control of the control instructions in the micro code program of the processor, these predefined DRAM control operations can be arranged in sequence to give almost any type of DRAM access, such as a read, write, page mode read, page mode write, page mode read write and page mode write read access.

This means that the micro code program itself freely can use page mode whenever appropriate, thus saving valuable time. In page mode, there is generally no need for a cache memory since the activated row will act as "cache memory".

Other advantages offered by the present invention will be appreciated upon reading of the below description of the embodiments of the invention.

BRIEF DESCRIPTION OF THE DRAWINGS

The invention, together with further objects and advantages thereof, will be best understood by reference to the following description taken together with the accompanying drawings, in which:

Fig. 1 is a schematic block diagram of a computer system according to a preferred embodiment of the invention;

Fig. 2 is a schematic block diagram of an address register and an address multiplexer used by the invention;

Fig. 3 is a schematic diagram illustrating a practical implementation of the interface between an arithmetic unit, and an address register and a data-to-memory register according to an alternative embodiment of the invention;

Fig. 4 is a schematic timing diagram of a number of DRAM control operations according to a preferred embodiment of the invention;

Fig. 5 is a schematic state diagram illustrating how the DRAM control operations can be arranged into DRAM accesses according to a preferred embodiment of the invention;

Fig. 6 is a schematic diagram of the DRAM control logic in a DRAM controller according to a preferred embodiment of the invention; and

Figs. 7A-6E are schematic timing diagrams of different DRAM accesses according to a preferred embodiment of the invention.

DETAILED DESCRIPTION OF EMBODIMENTS OF THE INVENTION

Throughout the drawings, the same reference characters will be used for corresponding or similar elements.

Fig. 1 is a schematic block diagram of a computer system according to a preferred embodiment of the invention. The computer system basically comprises a central processing unit (CPU) 10, a dynamic random access memory (DRAM) 60 and a data bus 70.

The CPU 10, also referred to as the processor, comprises a processor control unit 20, local storage units 30, an arithmetic unit 40 and a DRAM controller 50. The CPU 10 is controlled by micro code instructions stored in a micro code program 22. The micro code program 22 cooperates with a sequence control unit 24 that determines the address to the next micro code instruction in the micro code program. The arithmetic unit 40 is provided to perform arithmetical/logical operations on data from the DRAM 60, and it interfaces the DRAM directly (via the data bus 70). A cache memory is not required.

The DRAM controller 50 is connected between the DRAM 60 and the control unit 20 of the CPU 10, and controls the flow of data in and out of the DRAM. The DRAM controller 50 generates address signals ROW/COL and various control signals WEn, CASn, RASn, OEn (the suffix n indicates that the signals are active low) in a predetermined sequence under the control of the control unit 20, and applies these signals to the DRAM 60, thereby controlling the operation of the DRAM. The DRAM controller 50 comprises control logic 52, an address register 54 for storing address information received from the micro code program 22 via the data bus 70 and an address multiplexer 56 controlled by two complementary signals COL and COLn for applying the row address or the column address to the DRAM 60.

A general idea according to the invention is to integrate the DRAM access control into the micro code instructions of the micro code program 22. Each micro code instruction includes a control instruction that is used in controlling the operation of the DRAM controller 50. Furthermore, for each DRAM access, the operation of the DRAM controller 50 is divided into a sequence of a predetermined number of DRAM control operations. The DRAM controller 50 controls access to the DRAM by executing, for each access, a sequence of DRAM control operations in response to a corresponding sequence of control instructions included in the micro code instructions of the micro code program 22.

Each control instruction in the sequence of control instructions from the micro code program 22 to the DRAM controller 50 determines which one of a number of predefined DRAM control operations that the DRAM controller 50 will execute. By forwarding appropriate control instructions from the micro code program 22, the sequence of DRAM control operations can be customized to give different types of DRAM accesses, such as read, write, page mode read, page mode write, page mode read write and page mode write read accesses.

It is important to understand that each control instruction is formed by a predetermined part of a micro code instruction. The remaining part of the micro code instruction is used for other purposes such as determining address information for the DRAM or executing high-level instructions collected from the DRAM.

For completeness, the address register 54, and the address multiplexer 56 are shown in more detail in Fig. 2. The address register 54 contains three address portions, ADL (Address Low), ADH (Address High) and ADP (Address Page). The row address to the DRAM is stored in ADP and ADH in bits 10-18, and the column address to the DRAM is stored in ADH and ADL in bits 1-9. The address multiplexer 56 comprises two AND-gates followed by an OR-gate. The first AND-gate receives the column address from ADH and ADL, and the COL signal. The second AND-gate receives the row address from ADP and ADH, and the COLn signal. If COLn is inactive, i.e. high, the row address is output by the OR-gate. However, if COLn is active, i.e. low, the column address is output by the OR-gate.

Fig. 3 is a schematic diagram illustrating a practical implementation of the interface between the arithmetic unit 40 and the address register 54 and a data-to-memory register 84, according to an alternative embodiment of the invention. Data from a number of data sources, one of which is the DRAM, is recieved by a data source selector 72, the output of which forms a bus interface DBUS to the arithmetic unit 40, the address register 54 and a data-to-memory (DTM) register 84. The output of the arithmetic unit 40 forms a bus interface YBUS to the ADL register and to the DTM 84. The DTM 84 also receives a control signal DTM_ENABLE which enables data in DTM to be forwarded to the DRAM. The arithmetic unit 40 is responsive to micro code from the micro program (cf. Fig. 1). As can be seen from Fig. 3, ADL and DTM can be loaded either from the DBUS or the YBUS.

For a better understanding of the invention, the invention will now be explained with reference to an illustrative example of a number of predefined DRAM control operations.

Fig. 4 is a schematic timing diagram of a number of DRAM control operations according to a preferred embodiment of the invention. In this particular example, a 66 MHz main oscillator is utilized, and the main clock signal for the micro code instructions is designated by EXEC. For slanting edges, the position depends on the duty cycle of the 66 MHz oscillator, varying between 46% to 54%. The time period between vertical edges, as well as between slanting edges is 15 ns, and the time period between a vertical edge and a slanting edge is then between 6.9 ns and 8.1 ns. Furthermore, the cycle time of each micro code instruction is adjustable in such a way that each micro code instruction can take one of two different values for the micro instruction cycle time; 30 ns or 45 ns. This is indicated by dotted lines in Fig. 4. In Fig. 4, all signals are active low and this is indicated by the suffix n; SIGNALn. The DRAM control operations illustrated in Fig. 4 are R (Start Read), W (Start Write), H (Hold) and E (End Row Access), all of which will be explained below.

The R-operation (Start Read):

RASn is pulled down 15 ns after the start of the current micro code instruction, if RASn is not low already;

CASn is pulled up 7.5 ns before the end of the current micro code instruction, if CASn is not high already, and 7.5 ns after the start of the next micro code instruction, CASn is pulled down;

WEn is pulled up 7.5 ns before the end of the current micro code instruction, if WEn is not high already; and

COLn is pulled down 7.5 ns before the end of the current micro code instruction, if COLn is not low already.

The W-operation(Start Write):

RASn is pulled down 15 ns after the start of the current micro code instruction, if RASn is not low already;

CASn is pulled up 7.5 ns before the end of the current micro code instruction, if CASn is not high already, and 7.5 ns after the start of the next micro code instruction, CASn is pulled down;

WEn is pulled down 15 ns before the end of the current micro code instruction, if WEn is not low already; and

COLn is pulled down 7.5 ns before the end of the current micro code instruction, if COLn is not low already.

The H-operation (Hold):

CASn is pulled up 7.5 ns before the end of the current micro code instruction, if CASn is not high already;

WEn is pulled up 7.5 ns before the end of the current micro code instruction, if WEn is not high already;

The E-operation (End):

RASn is pulled up 15 ns after the start of the current micro code instruction, if RASn is not high already;

CASn is pulled up 7.5 ns before the end of the current micro code instruction, if CASn is not high already;

WEn is pulled up 7.5 ns before the end of the current micro code instruction, if WEn is not high already; and

COLn is pulled up 7.5 ns before the end of the current micro code instruction, if COLn is not high already.

The DRAM controller 50 is instructed by a sequence of control instructions, one control instruction in each micro code instruction of the micro code program 22, to execute a corresponding sequence of DRAM control operations. In the case of a total of four different DRAM control operations, two control bits in each control instruction are sufficient to select one of the four operations. In the present embodiment, the control instruction field, designated by MEMCP, in the micro code instructions is designed to distinguish between the different DRAM control operations in the following way:

Table I

MEMCP field	DRAM control operation
0 0	R
0 1	W
1 0	H
1 1	E

Fig. 5 is a schematic state diagram illustrating how the DRAM control operations can be arranged into DRAM accesses according to a preferred embodiment of the invention. Either an R-operation or a W-operation can follow an E-operation. An R-operation can be followed by an H-operation, which in turn can be followed by any type of operation. An R-operation or an H-operation can follow a W-operation. All operations can be iterated, i.e. repeated as many times as desired.

A DRAM access may be initiated by performing an R-operation, after an E-operation. During this microinstruction cycle, the row address must be valid in ADP and ADH. The next DRAM control operation must be of type R or H, and during this microinstruction cycle, the column address must be valid in ADH and ADL. For an R-operation that does not initiate the DRAM access, the row address need not be valid. In the subsequent microinstruction cycle when a further DRAM control operation is started in accordance with the state diagram of Fig. 5, data corresponding to the applied DRAM address, i.e. the row and column address, is available and can be used by the micro code instruction. Data can be used by subsequent micro code instructions as well, if not replaced by data from a new read-out.

A DRAM access may also be initiated by performing a W-operation, after an E-operation. During this microinstruction cycle, the row address must be valid in ADP and ADH. The next DRAM control operation must be of type W, R or H, and during this microinstruction cycle, the column address must be valid in ADH and ADL, and the data to be written to the applied DRAM address must be valid in the data-to-memory register. For a W-operation that does not initiate the DRAM access, the row address need not be valid.

The H-operation can be inserted anywhere as a “no operation” to temporarily put the DRAM access cycle on hold, for example when the steps required for the DRAM access are not sufficient to accomplish the work to be done by the micro code program as specified by other fields of the micro code instructions. In this case, the requirement on valid column address in ADH and ADL is connected to the previous DRAM control operation being of type R or W. Similarly, the requirement on valid data in the DTM register always appears one step after a W-operation.

The H-operation has been chosen as the default type of operation, since it can be inserted anywhere. The E-type operation is also allowed as a “no operation”, but only when the DRAM is idle, i.e. in any sequence between an E-operation and the next R- or W-operation.

For example, a normal DRAM read access can be made in three microinstruction cycles. During each micro code instruction, the control instruction included in the micro code instruction instructs the DRAM controller 50 to perform a DRAM control operation. For a read access, a sequence of the control operations R, H and E is normally performed. At the first DRAM control operation, R, the row address is valid in the address register 54 and the row address is strobed into the DRAM 60 by the control signal RASn. In the microinstruction cycle, in which R-operation is being initiated, the micro code instruction also loads the column address into the address register 54. At the second DRAM control operation, H, the column address, loaded into the address register in the previous microinstruction cycle, is strobed into the DRAM by the control signal CASn. At the third DRAM control operation, E, a data-from-memory latch (not shown) presents data from the DRAM. This data is available to be used by the current micro code instruction or subsequent micro code instructions.

In the following, examples of DRAM control operation sequences for a number of different DRAM accesses are shown in Table II.

Table II

DRAM ctrl op.	Requirements/status relevant for the Contents of other parts of the microinstruction	Comment
Read:		
(R)	Row addr in ADH	Begin row access, start read
(H)	Col addr in ADL	
(E)	Data available	Read, end row access
Read – page mode:		
(R)	Row addr in ADH	Begin row access, start read (#1)
(R)	Col addr (#1) in ADL	Start read (#2)
(R)	Col addr (#2) in ADL, Data (#1) available	Read (#1), start read (#3)
(R)	Col addr (#3) in ADL, Data (#2) available	Read (#2), start read (#4)
(R)	Col addr (#4) in ADL, Data (#3) available	Read (#3), start read (#5)
(H)	Col addr (#5) in ADL, Data (#4) available	Read (#4)
(E)	Data (#5) available	Read (#5), end row access
Write:		
(W)	Row addr in ADH	Begin row access, start write
(H)	Col addr in ADL, Data from DTM	Write
(E)		End row access
Write -- page mode:		
(W)	Row addr in ADH	Begin row access, start write (#1)
(W)	Col addr (#1) in ADL Data (#1) in DTM	Write (#1), start write (#2)
(W)	Col addr (#2) in ADL, Data (#2) in DTM	Write (#2), start write (#3)
(W)	Col addr (#3) in ADL, Data (#3) in DTM	Write (#3), start write (#4)
(W)	Col addr (#4) in ADL, Data (#4) in DTM	Write (#4), start write (#5)
(H)	Col addr (#5) in ADL, Data (#5) in DTM	Write (#5)
(E)		End row access

Read first, then write:

(R)	Row addr in ADH	Begin row access, start read
(C)	Col addr (#1) in ADL	
(W)	Data (#1) available	Read, start write
(H)	Col addr (#2) in ADL, Data (#2) in DTM	Write
(E)		End row access

Page mode read first, then page mode write:

(R)	Row addr in ADH	Begin row access, start read (#1)
(R)	Col addr (#1) in ADL	Start 2nd read (#2)
(R)	Col addr (#2) in ADL, Data (#1) available	Read (#1), start read (#3)
(C)	Col addr (#3) in ADL, Data (#2) available	Read (#2)
(W)	Data (#3) available	Read (#3), start write (#4)
(W)	Col addr (#4) in ADL, Data (#4) in DTM	Write, start next write (#5)
(W)	Col addr (#5) in ADL, Data (#5) in DTM	Write, start next write (#6)
(H)	Col addr (#6) in ADL, Data (#6) in DTM	Last write
(E)		End row access

Write first, then read:

(W)	Row addr in ADH	Begin row access, start write
(R)	Col addr (#1) in ADL Data (#1) in DTM	Write, start read (#2)
(H)	Col addr (#2) in ADL,	
(E)	Data (#2) available	Read (#2), end row access

Page mode write first, then page mode read:

(W)	Row addr in ADH	Begin row access, start 1st write
(W)	Col addr (#1) in ADL Data (#1) in DTM	1st write, start 2nd write
(W)	Col addr (#2) in ADL, Data (#2) in DTM	2nd write, start 3rd write
(R)	Col addr (#3) in ADL, Data (#3) in DTM	3rd write, start read (#4)
(R)	Col addr (#4) in ADL,	Start read (#5)
(R)	Col addr (#5) in ADL, Data (#4) available	Read (#4), start read (#6)
(H)	Col addr (#6) in ADL, Data (#5) available	Read (#5)
(E)	Data (#6) available	Read (#6), end row access

"Row addr in ADH" means:

Contents of ADP and (mainly) ADH during this micro code instruction (loaded into the registers by earlier micro code instructions) will be fed to the DRAM and used as row address for all DRAM accesses until a micro code instruction with the "E" code in the MEMCP field has been executed. Loading ADP or ADH in this micro code instruction is allowed but will not have any effect on these memory accesses.

"Col addr in ADL" means:
Contents of ADH and (mainly) ADL during this micro code instruction (loaded into the registers by earlier micro code instructions) will be fed to the DRAM and used as column address for this DRAM access (which was started by R or W in the previous micro code instruction). Loading ADH or ADL in this micro code instruction is allowed but will not have any effect on this memory access.

"Data in DTM" means:
Contents of DTM register during this micro code instruction (loaded into the register by earlier micro code instruction) will be fed to the DRAM and used as data for this DRAM access (which was started by W in the previous micro code instruction). Loading DTM in this micro code instruction is allowed but will not have any effect on this memory access.

"Data available" means:
The DFM (Data From Memory) latch now presents data from DRAM, available to be used by this micro code instruction. The contents will continue to be available until changed by a new read access, marked here by a new "Data available" note.

"#1", "#2", etc. means:
These numbers mark different memory accesses, to show the correspondence between column addresses and data bytes.

As can be seen, each DRAM control operation more or less corresponds to a subcycle of a complete DRAM access.

The micro code integrated access control is particularly advantageous for processors that have to determine the DRAM address in two steps; for example an 8-bit processor interfacing a DRAM with 16-bit addressing. The DRAM access can be initiated before the complete DRAM address has been determined.

Fig. 6 is a schematic diagram of the DRAM control logic in a DRAM controller according to a preferred embodiment of the invention. The DRAM control logic 52 is implemented by using a number of conventional gates (AND-gates AN, NOR-gates NR and NAND-gates ND), flip-flops (DFF and JKFF) and inverters (I). The control logic 52 is responsive to the control instructions in the MEMCP field of the micro code instructions in the micro code program 22. For each micro code instruction received from the micro code program 22, the control logic 52 generates DRAM control signals for a predetermined subcycle of a DRAM access in response to the control bits in the MEMCP field.

In this particular embodiment, the two control bits in the MEMCP field of the current micro code instruction are applied to the gates AN, NR, AN following the inverter I, to generate a number of signals E, R and W which in turn are fed to different gates and flip-flops in the control logic 52. In addition, the first MEMCP control bit is applied to a JK flip-flop, and two NAND-gates ND. The second MEMCP control bit is furthermore applied to a NOR-gate NR. The signal A is the output of a flip-flop ALLRAS used for setting a RAS signal to all memory circuits in the DRAM during a refresh cycle. The signal 66 is the output signal of the 66 MHz oscillator, and EXEC is the main clock signal for the micro code instructions.

In response to the MEMCP control bits, the DRAM control signals RASn, CASn, WEn, COL and COLn are generated. If a 16-bit DRAM is used, the CAS signal may be utilized together with the ADDR LSB from the address register 54 and a further control signal 16BITn to selectively generate UCASn and LCASn signals. UCASn stands for Column Address Strobe/Upper Byte Control Bit, and LCASn stands for Column Address Strobe/Lower Byte Control Bit. The signals UCASn and LCASn are used in a conventional manner to access respective parts of the 16-bit DRAM.

Furthermore, a flip-flop DFF generates a data-to-memory enable (DTM_ENABLE) signal in response to the signals W and EXEC. The

DTM_ENABLE signal controls the data-to-memory register. As can be seen from Fig. 6, the control logic 52 also generates a hold-data-from-memory (HOLD_DFM) signal that controls the data-from-memory latch (not shown).

Further detailed information on DRAMs and DRAM control can be found, for example in the data book *MOS Memory DRAM (Byte/Word Wide)* from Toshiba (1995), and particularly the DRAM specification TC51V4265DJ/DFT-60, -70 on pages 1604-1634 thereof.

Figs. 7A-E are schematic timing diagrams of different DRAM accesses according to a preferred embodiment of the invention.

Fig. 7A is a schematic timing diagram of a page mode read access according to a preferred embodiment of the invention. As detailed in Table II above, the sequence of DRAM control operations includes a predetermined number of R-operations followed by an H-operation and an E-operation.

In the first microinstruction cycle, 30 ns long, the micro code instruction determines a row address ROW and loads it into ADH.

In the second microinstruction cycle, which is extended to 45 ns, an R-operation is executed and a first column address COL1 is determined by the micro code instruction and loaded into ADL. The row address ROW is strobed into the DRAM, activating the selected row.

In the third microinstruction cycle which also may be extended to 45 ns depending on the cycle time of the DRAM used, an R-operation is executed once again and a second column address COL2 is determined by the micro code instruction and loaded into ADL. The column address COL1 is strobed into the DRAM.

In the fourth microinstruction cycle, an R-operation is executed again executed and a third column address COL3 is determined by the micro code instruction and loaded into ADL. The column address COL2 is strobed into the DRAM. Now, the DFM latch, controlled by the HOLD_DFM signal, presents the first data DATA1, associated with the DRAM address (ROW, COL1), from the DRAM. The data source DSOURCE is the DFM latch holding DATA1.

In the fifth microinstruction cycle, an H-operation is executed. The column address COL3 is strobed into the DRAM. The DFM latch now presents the next data DATA2, associated with the DRAM address (ROW, COL2), from the DRAM; DSOURCE is DFM(DATA2).

In the last microinstruction cycle of the page mode read access illustrated in Fig. 7A, an E-operation is executed. The DFM latch now presents the DATA3, associated with the DRAM address (ROW, COL3), from the DRAM; DSOURCE is DFM(DATA2).

Fig. 7B is a schematic timing diagram of a page mode read write access according to a preferred embodiment of the invention. The sequence of DRAM control operations for a read write access is detailed in Table II above. First, an R-operation is executed, followed by an H-operation. Next, a W-operation is executed and the data-to-memory (DTM) register is loaded. The column address for writing is loaded into ADL in this microinstruction cycle or in the previous cycle. At the same time, the DFM latch presents the data DATA1 from the DRAM. In the next microinstruction cycle, an H-operation is executed and the column address (COL2) for writing is strobed into the DRAM and the data (DATATOMEM1) in DTM is written into the DRAM. The access is ended by an E-operation.

Fig. 7C is a schematic timing diagram of a page mode write access according to a preferred embodiment of the invention. The sequence of DRAM control operations for a page mode write access is detailed in Table II above. A page mode write access is similar to a page mode read access. W-operations are used instead of R-operations. The data to be written into the DRAM is loaded into the DTM register at least one microinstruction cycle before the actual writing into the DRAM.

Fig. 7D is a schematic timing diagram of a page mode write read access according to a preferred embodiment of the invention. The sequence of DRAM control operations for a page mode write read access is detailed in Table II above. It should be noted that the W-operation is followed directly by an R-operation, without the need for an H-operation in between.

Fig. 7E is a schematic timing diagram of CAS before RAS refresh cycle according to a preferred embodiment of the invention. The sequence of DRAM control operations for such a DRAM access cycle starts with an H-operation in which ALLRAS is set, followed by an E-operation. When ALLRAS is set, the signal A goes high, and the control logic 52 (Fig. 6) is affected accordingly. Next, a sequence of the DRAM control operations R, H and E are repeated a suitable number of times, and in the last repetition the H-operation includes a reset of ALLRAS. The access cycle is terminated by an E-operation. Alternatively, a sequence of the DRAM control operations H, E and R is repeated.

In the currently most preferred embodiment of the invention, the central processor is a complex instruction set computing (CISC) processor, and complex instructions are stored in the DRAM and executed by micro code instructions stored in the micro code program memory. The micro code program memory is preferably static, although nothing prevents it from being dynamic. In this case, the complex instructions, also referred to as high-level machine instructions, as well as the data are stored in the DRAM, and therefore the computer system may be referred to as a CISC with von Neumann architecture. When the micro code program has selected and activated a row in the DRAM, the micro code program has quick access to all the memory cells of the row, in any arbitrary order and with an arbitrary combination of read and write operations.

Alternatively, the arithmetical part of the processor is designed to have two DRAM-interfaces, one interface for the complex instructions and one for data. This would then enable a CISC with Harvard architecture.

However, the "micro code instructions" could be the instructions of a reduced instruction set computing (RISC) processor, so that the computer system would be referred to as a RISC-system with Harvard architecture; where instructions and data are stored in different memories.

It should be understood that the number of DRAM control operations is not limited to four as in the preferred embodiment of the invention. It is possible to use more or less than four DRAM control operations. The number of control bits in each micro code instruction will then be modified accordingly. If more than four DRAM control operations are used by the invention, each micro code instruction has to include more than two control bits. A single control bit can distinguish between two DRAM control operations.

The embodiments described above are merely given as examples, and it should be understood that the present invention is not limited thereto. Further modifications, changes and improvements which retain the basic underlying principles disclosed and claimed herein are within the scope and spirit of the invention.

CLAIMS

1. A method for controlling access to a dynamic random access memory (DRAM),
characterized in that said method comprises the step of performing, for each DRAM access, a sequence of a predetermined number of DRAM control operations in response to a corresponding sequence of control instructions included in micro code instructions of a processor (10).

2. The method according to claim 1,
characterized in that each micro code instruction includes a control instruction, formed by at least one control bit, controlling which one of a plurality of predefined DRAM control operations (R, W, H, E) to perform.

3. The method according to claim 2,
characterized in that said predefined DRAM control operations (R, W, H, E) are arrangeable to form said sequence of DRAM control operations such that a read access, a write access, a page mode read access, a page mode write access, a page mode read write access or a page mode write read access to said DRAM (60) is enabled.

4. The method according to any of the preceding claims,
characterized in that at least one control instruction in said sequence of control instructions temporarily puts the memory cycle of said DRAM (60) on hold.

5. The method according to any of the preceding claims,
characterized in that the duration of said DRAM control operations matches the cycle time of said micro code instructions.

6. The method according to claim 5,
characterized in that said method further comprises the step of selecting the cycle time of each micro code instruction from a number of different cycle times.

7. The method according to claim 2,
characterized in that a first one, referred to as an R-operation, of said predefined DRAM control operations includes the steps of:

selectively, if inactive, activating a row address strobe (RAS) signal to said DRAM;

selectively enabling a valid row address to be forwarded to said DRAM (60), and, a first predetermined period of time later, enabling a valid column address to be forwarded to said DRAM (60);

selectively, if active, deactivating a write enable (WE) signal to said DRAM; and

selectively, if active, deactivating a column address strobe (CAS) signal to said DRAM, and, a second predetermined period of time later, in the next microinstruction cycle, activating said CAS signal.

8. The method according to claim 2,
characterized in that a second one, referred to as a W-operation, of said predefined DRAM control operations includes the steps of:

selectively, if inactive, activating a row address strobe (RAS) signal to said DRAM;

enabling a valid row address to be forwarded to said DRAM (60), and, a first predetermined period of time later, enabling a valid column address to be forwarded to said DRAM;

selectively, if inactive, activating a write enable (WE) signal to said DRAM; and

selectively, if active, deactivating a column address strobe (CAS) signal to said DRAM, and, a second predetermined period of time later, in the next microinstruction cycle, activating said CAS signal.

9. The method according to claim 2,
characterized in that a third one, referred to as a H-operation, of said predefined DRAM control operations includes the steps of:

deactivating a column address strobe (CAS) signal to said DRAM (60); and

deactivating a write enable signal to said DRAM.

10. The method according to claim 2,
characterized in that a fourth one, referred to as an E-operation, of said predefined DRAM control operations includes the steps of:

deactivating a row address strobe (RAS) signal to said DRAM (60);

selectively, if active, deactivating a column address strobe (CAS) signal to said DRAM;

selectively, if active, deactivating a write enable (WE) signal to said DRAM; and

enabling a valid row address to be forwarded to said DRAM.

11. The method according to claim 7, 9 and 10,
characterized in that, for a read access to said DRAM (60), said sequence of DRAM control operations includes an R-operation, an H-operation and an E-operation, in that order.

12. The method according to claim 8, 9 and 10,
characterized in that, for a write access to said DRAM (60), said sequence of DRAM control operations includes a W-operation, an H-operation and an E-operation, in that order.

13. The method according to claim 7, 9 and 10,
characterized in that, for a page mode read access to said DRAM (60), said sequence of DRAM control operations includes a predetermined number of R-operations followed by an H-operation and an E-operation.

14. The method according to claim 8, 9 and 10, **characterized in that**, for a page mode write access to said DRAM (60), said sequence of DRAM control operations includes a predetermined number of W-operations followed by an H-operation and an E-operation.

15. A controller for a dynamic random access memory (DRAM), **characterized in that** said DRAM controller (50) is responsive to a sequence of control instructions for controlling access to said DRAM (60), each control instruction being formed by a predetermined part of a micro code instruction of a processor (10).

16. The DRAM controller according to claim 15, **characterized in that** said DRAM controller (50) controls access to said DRAM by performing a sequence of a predetermined number of DRAM control operations in response to said sequence of control instructions.

17. The DRAM controller according to claim 15, **characterized in that** each control instruction, formed by at least one control bit, controls which one of a plurality of predefined DRAM control operations (R, W, H, E) to perform.

18. The DRAM controller according to claim 16 or 17, **characterized in that** the duration of said DRAM control operations matches the cycle time of said micro code instructions.

19. The DRAM controller according to claim 18, **characterized in that** the cycle time of each micro code instruction is adjustable.

20. The DRAM controller according to claim 15, **characterized in that** at least one control instruction in said sequence of control instructions temporarily puts the memory cycle of said DRAM (60) on hold.

21. The DRAM controller according to claim 15, **characterized in that** the micro code instructions of said processor (10) are stored in a program memory (22) separated from said DRAM (60).

22. The DRAM controller according to claim 15, **characterized in that** said DRAM controller (50) is responsive to address information, determined by a number of micro code instructions of said processor (10), for addressing said DRAM (60).

23. The DRAM controller according to claim 15, **characterized in that** the micro code instructions of said processor (10) are the instructions of a reduced instruction set computing (RISC) processor.

24. A computer system having a processor (10), a dynamic random access memory (DRAM) cooperating with said processor, and a controller (50) for said DRAM (60), **characterized in that** said DRAM controller (50) is responsive to a sequence of control instructions from said processor (10) for controlling access to said DRAM (60), each control instruction being formed by a predetermined part of a micro code instruction of said processor (10).

25. The computer system according to claim 24, **characterized in that** said DRAM controller (50) controls access to said DRAM (60) by performing a sequence of DRAM control operations in response to said sequence of control instructions.

26. The computer system according to claim 24, **characterized in that** said processor (10) and said DRAM (60) are provided on the same circuit board.

27. The computer system according to claim 24, **characterized in that** said processor (10) is a complex instruction set computing (CISC) processor, and complex instructions are stored in said DRAM (60) and executed by micro code instructions stored in a program memory (22) in said processor (10).

ABSTRACT OF THE DISCLOSURE

The invention relates to a dynamic random access memory, and a method and circuit for controlling access to the DRAM. The access control for the DRAM (60) is intimately associated with the micro code instructions of a processor (10) connected to the DRAM. The access control is integrated into the micro code program (22) of the processor, and each micro code instruction includes a control instruction used in controlling the operation of a DRAM controller (50). More particularly, the DRAM controller (50) controls access to the DRAM (60) by executing, for each DRAM access, a sequence of DRAM control operations in response to a corresponding sequence of control instructions included in the micro code instructions of the processor. Preferably, the control instruction included in each micro code instruction only constitutes a part of the micro code instruction, and the remaining part of the micro code instruction is used by the processor for other purposes, such as determining address information for the DRAM and processing the data that is transferred between processor and DRAM.

(Fig. 1)

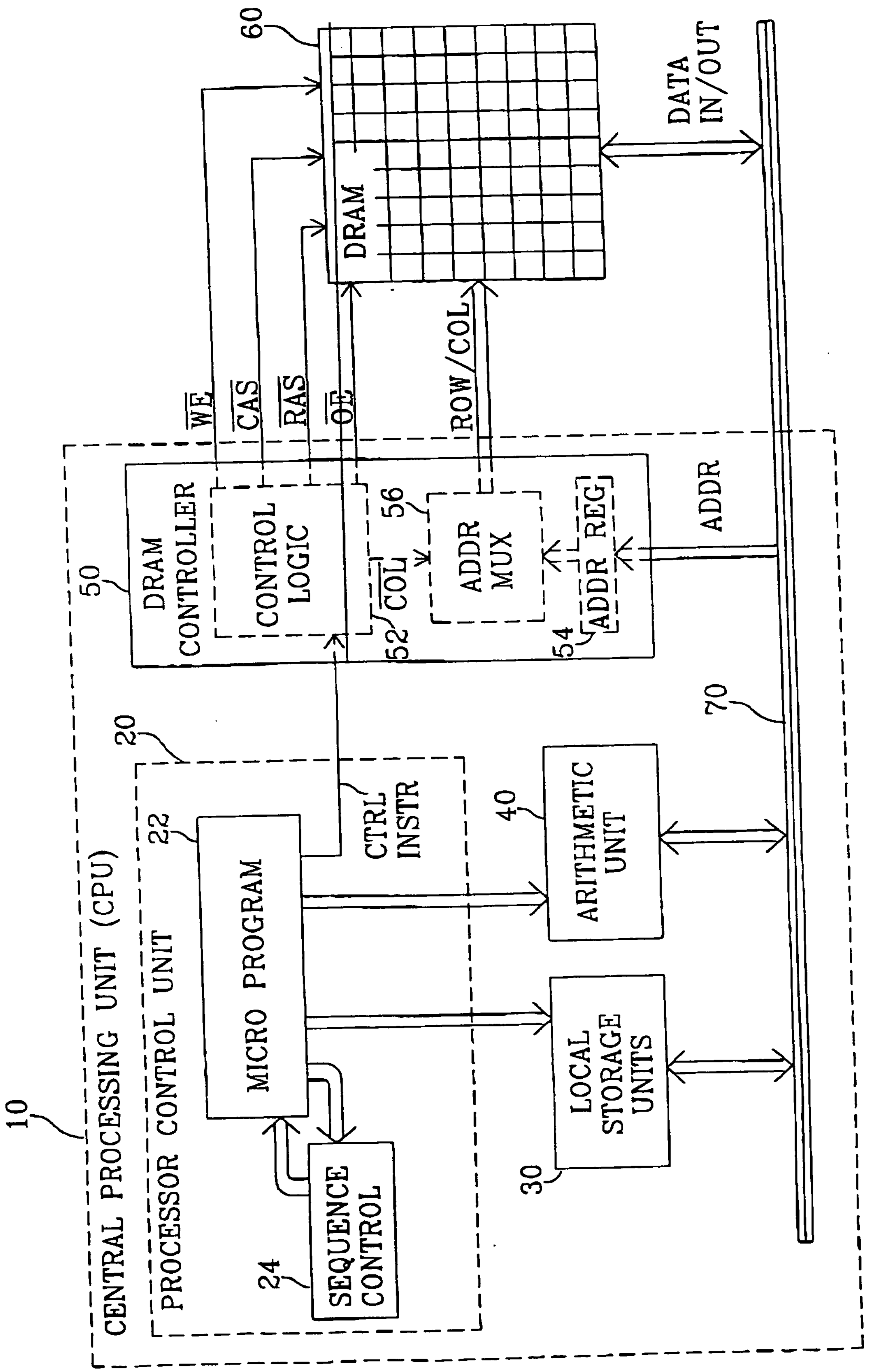
10
CENTRAL PROCESSING UNIT (CPU)
20
PROCESSOR CONTROL UNIT
22
MICRO PROGRAM
24
SEQUENCE CONTROL
CTRL INSTR
30
LOCAL STORAGE UNITS
40
ARITHMETIC UNIT
50
DRAM CONTROLLER
CONTROL LOGIC
52
COL
56
ADDR MUX
54
ADDR REG
ADDR
70
WE
CAS
RAS
OE
ROW/COL
60
DRAM
DATA IN/OUT

Fig. 1

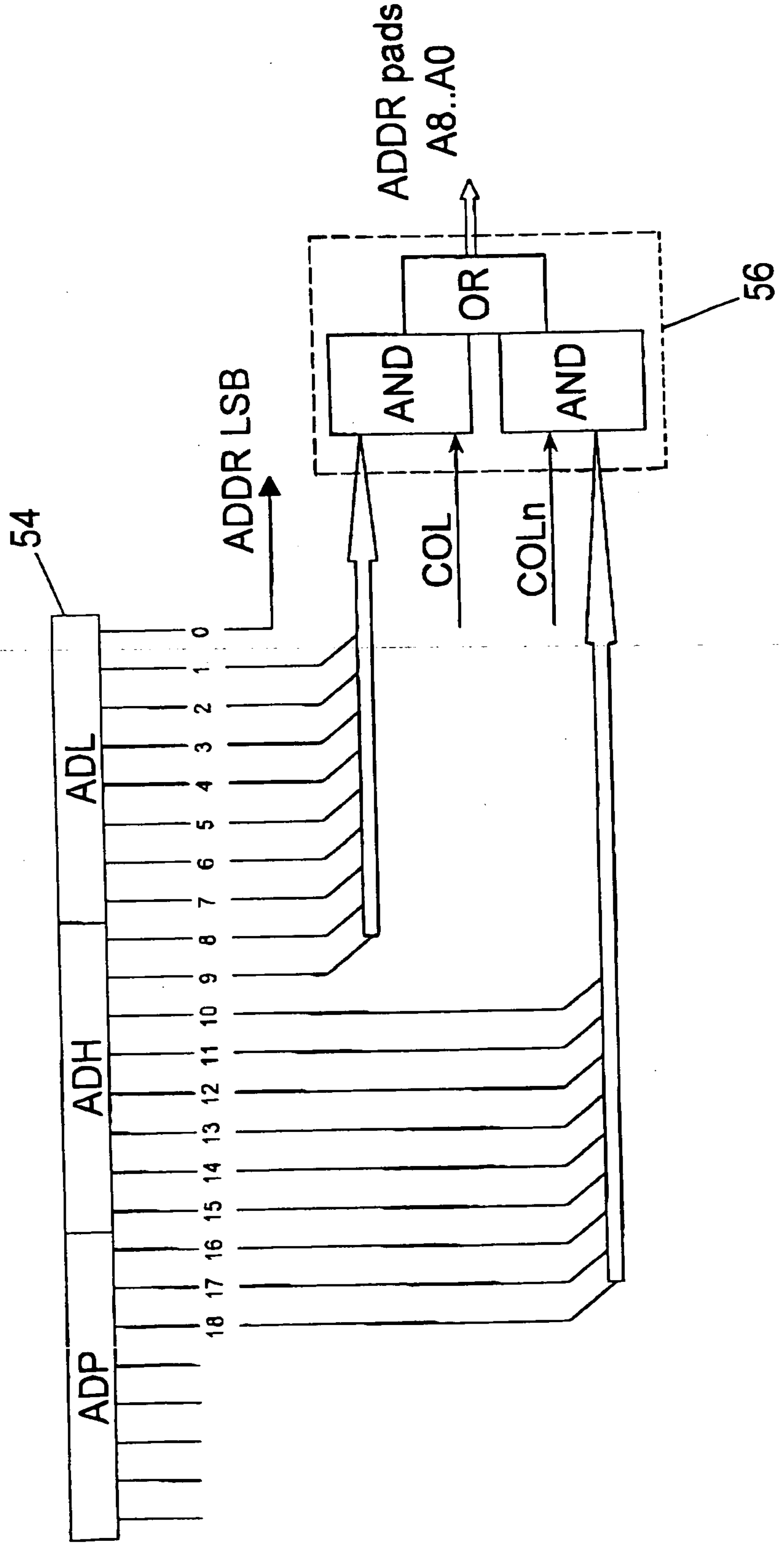
ADDR pads
A8..A0
OR
AND
AND
56
ADDR LSB
54
COL
COLn
ADL
ADH
ADP
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18

Fig. 2

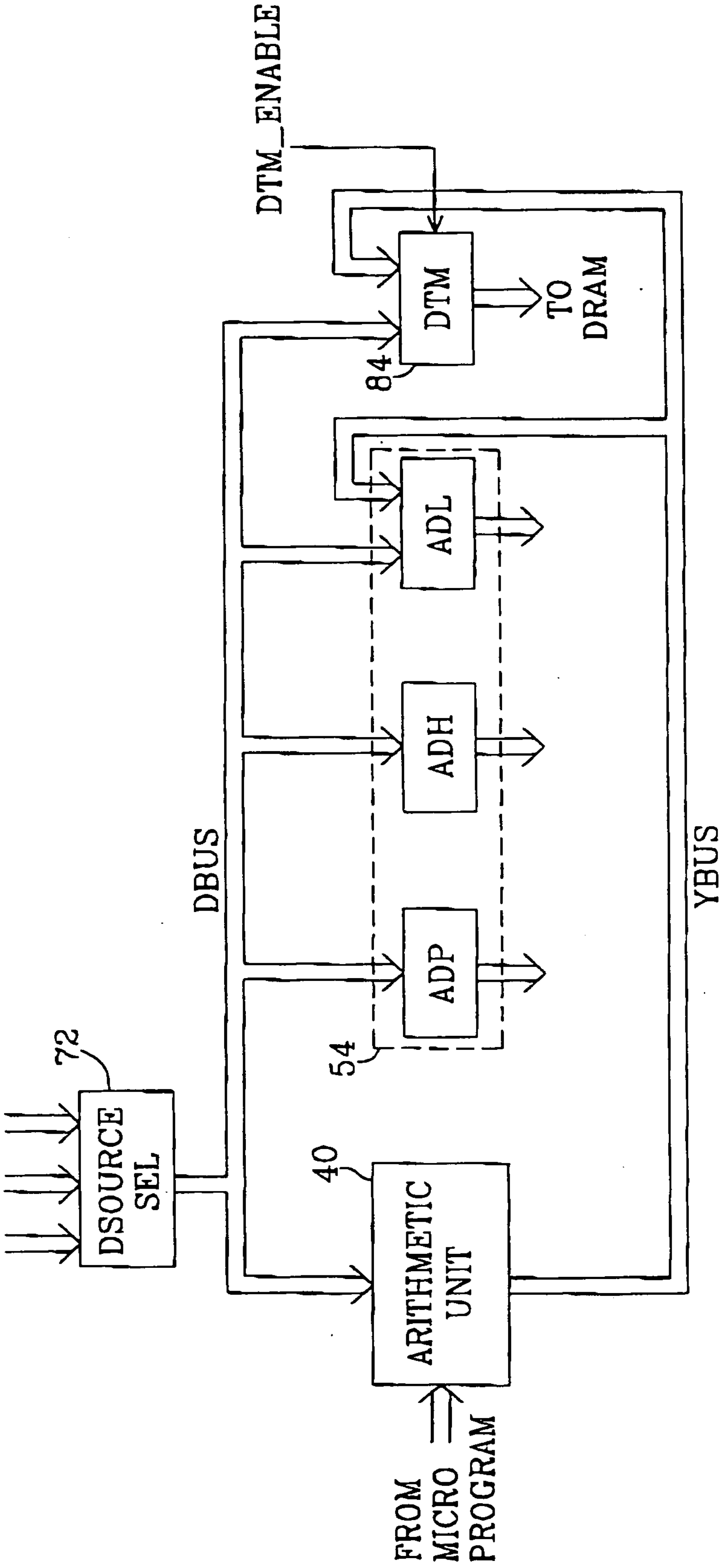
DTM_ENABLE
DTM
TO DRAM
84
ADL
ADH
ADP
54
DBUS
YBUS
DSOURCE SEL
72
40
ARITHMETIC UNIT
FROM MICRO PROGRAM

Fig. 3

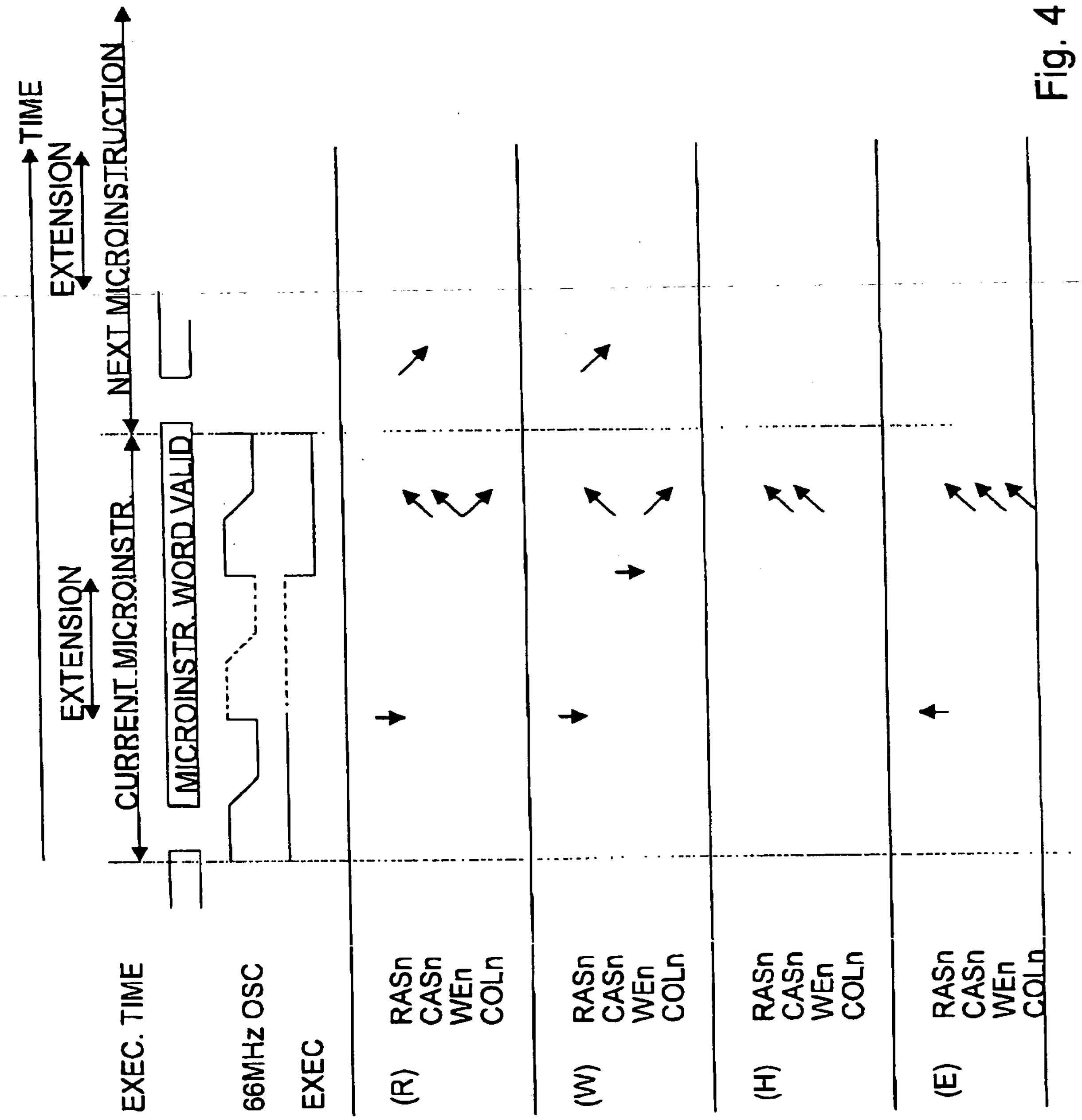
TIME
EXTENSION
NEXT MICROINSTRUCTION
EXTENSION
CURRENT MICROINSTR.
MICROINSTR. WORD VALID
EXEC. TIME
66MHz OSC
EXEC
(R)
RASn
CASn
WEn
COLn
(W)
RASn
CASn
WEn
COLn
(H)
RASn
CASn
WEn
COLn
(E)
RASn
CASn
WEn
COLn

Fig. 4

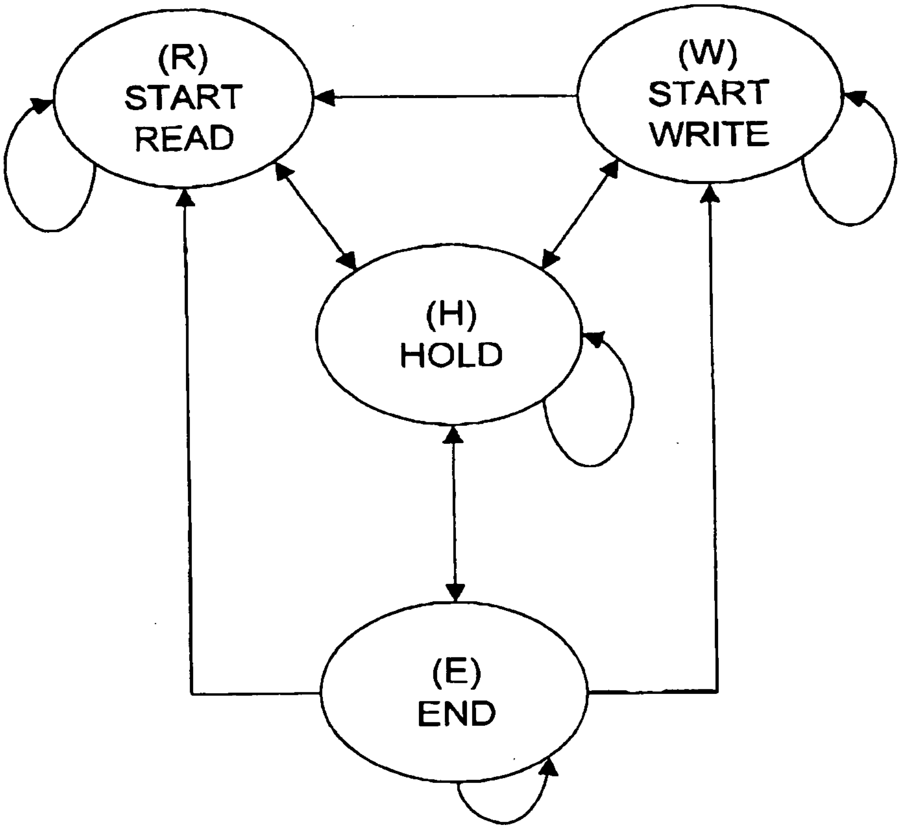

Fig. 5

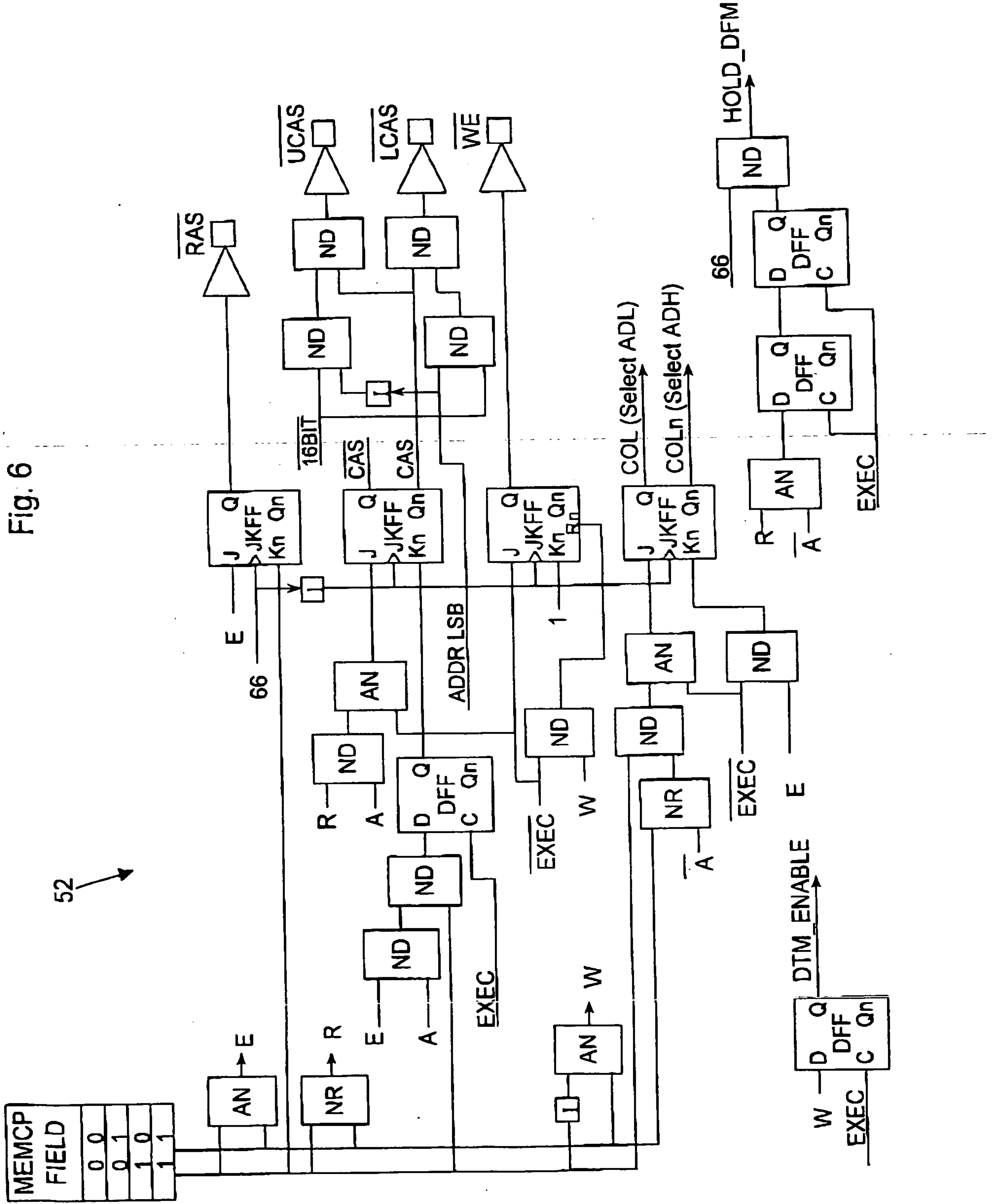

Fig. 6

52
MEMCP FIELD
0 0
0 1
1 0
1 1
AN
NR
E
R
ND
ND
ND
DFF
D
C
Q
Qn
AN
JKFF
J
Kn
RAS
UCAS
LCAS
WE
16BIT
CAS
CAS
ADDR LSB
66
E
A
W
EXEC
1
Rn
COL (Select ADL)
COLn (Select ADH)
HOLD_DFM
DTM ENABLE
1

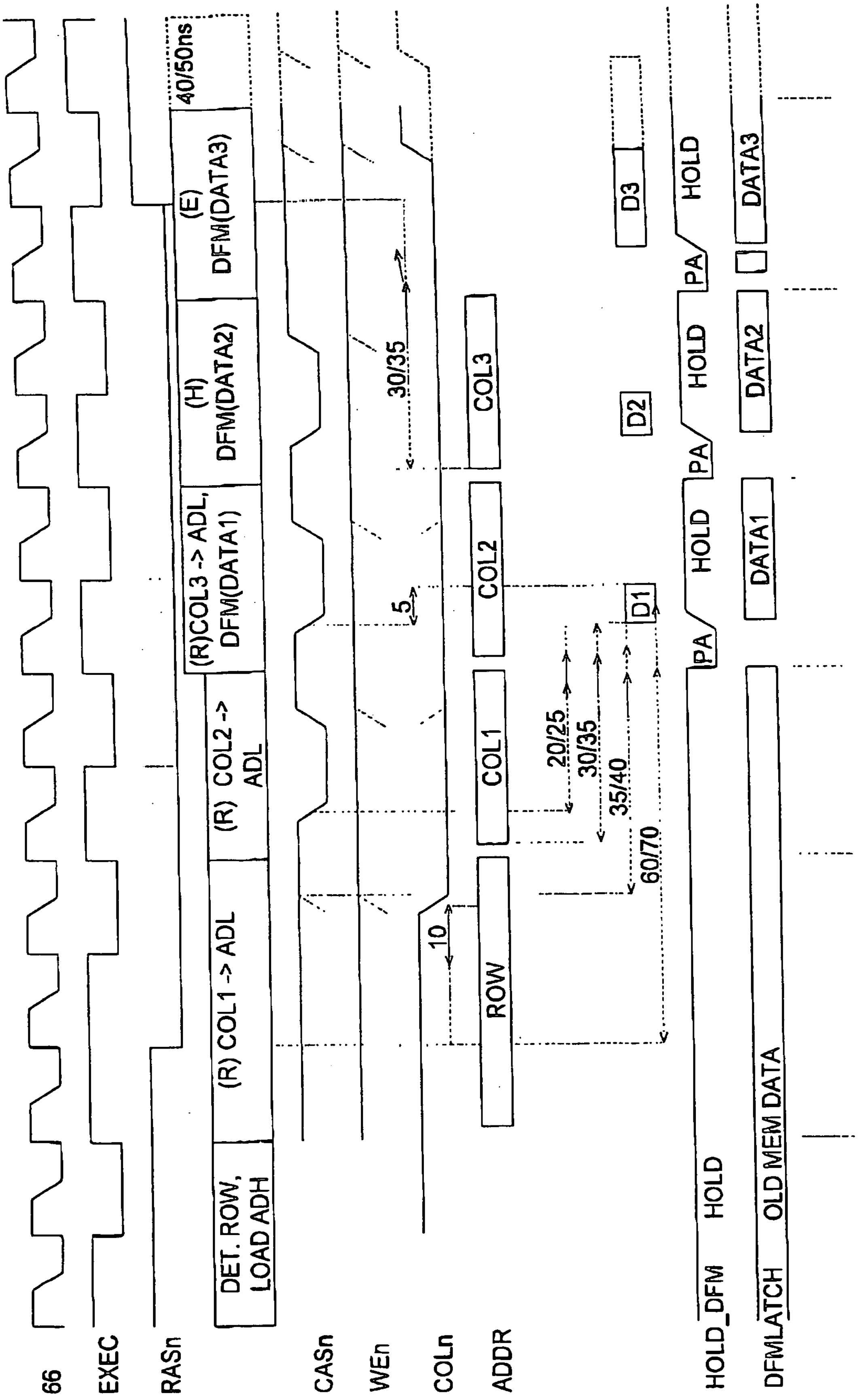
66
EXEC
RASn
CASn
WEn
COLn
ADDR
HOLD_DFM
DFMLATCH
40/50ns
DET. ROW, LOAD ADH
(R) COL1 -> ADL
(R) COL2 -> ADL
(R)COL3 -> ADL, DFM(DATA1)
(H) DFM(DATA2)
(E) DFM(DATA3)
ROW
COL1
COL2
COL3
10
5
30/35
20/25
30/35
35/40
60/70
D1
D2
D3
HOLD
PA
HOLD
PA
HOLD
PA
HOLD
OLD MEM DATA
DATA1
DATA2
DATA3

Fig. 7A

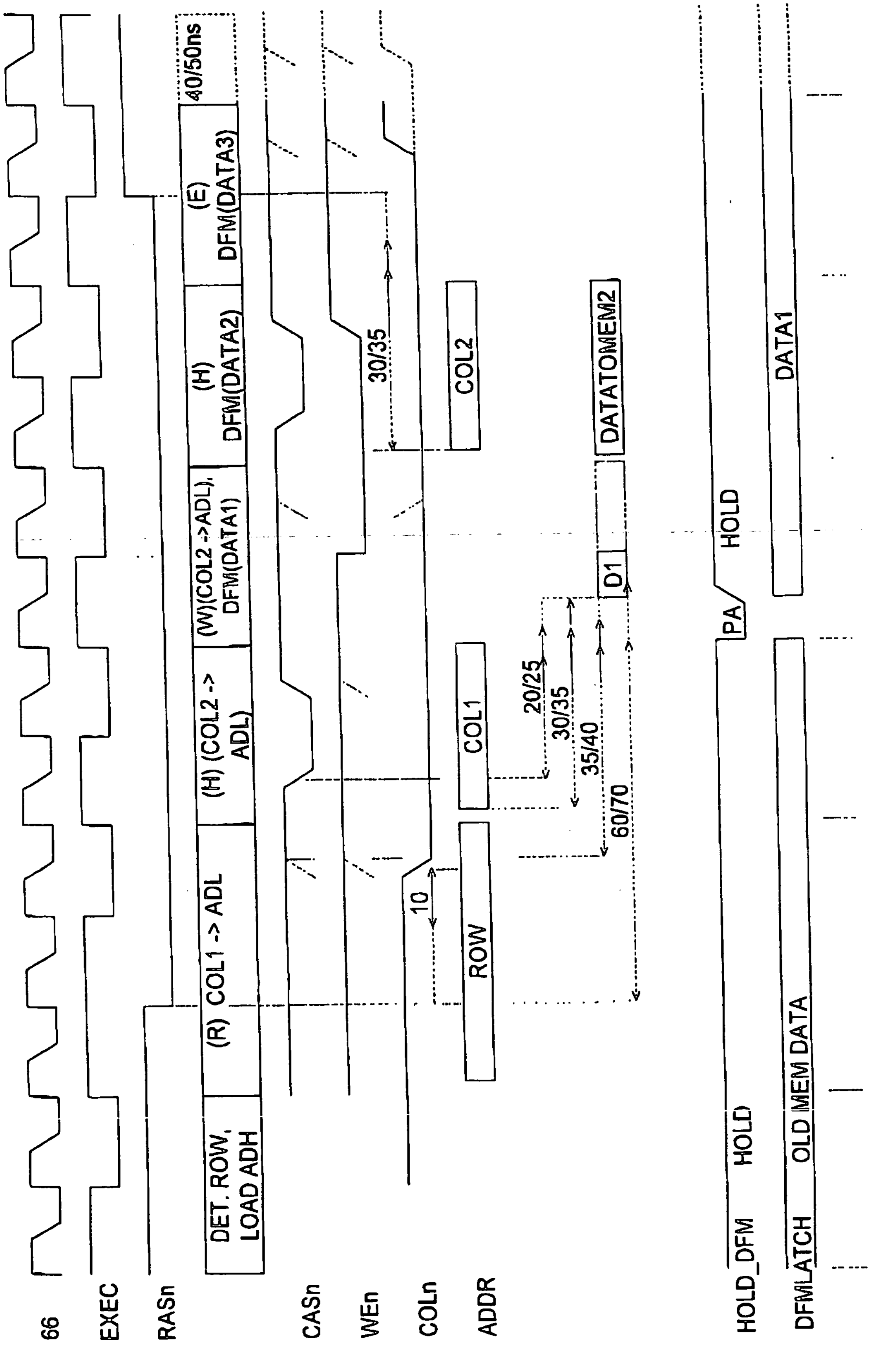
66
EXEC
RASn
CASn
WEn
COLn
ADDR
HOLD_DFM
DFMLATCH
DET. ROW, LOAD ADH
(R) COL1 -> ADL
(H) (COL2 -> ADL)
(W)(COL2 ->ADL), DFM(DATA1)
(H) DFM(DATA2)
(E) DFM(DATA3)
40/50ns
30/35
COL2
DATATOMEM2
D1
COL1
20/25
30/35
35/40
60/70
10
ROW
HOLD
PA
HOLD
DATA1
OLD MEM DATA

Fig. 7B

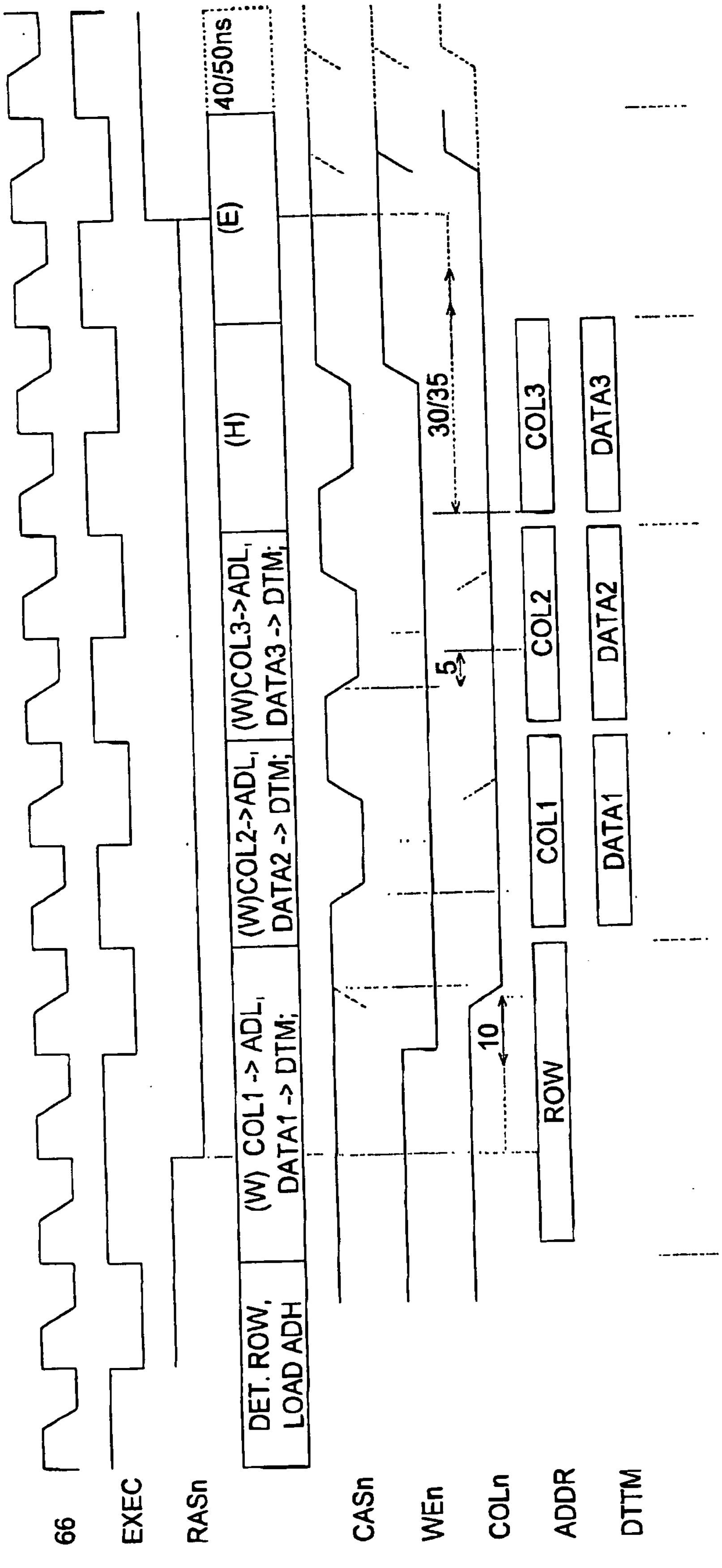
66
EXEC
RASn
CASn
WEn
COLn
ADDR
DTTM
DET. ROW, LOAD ADH
(W) COL1 -> ADL, DATA1 -> DTM;
(W)COL2->ADL, DATA2 -> DTM;
(W)COL3->ADL, DATA3 -> DTM;
(H)
(E)
40/50ns
30/35
5
10
ROW
COL1
COL2
COL3
DATA1
DATA2
DATA3

Fig. 7C

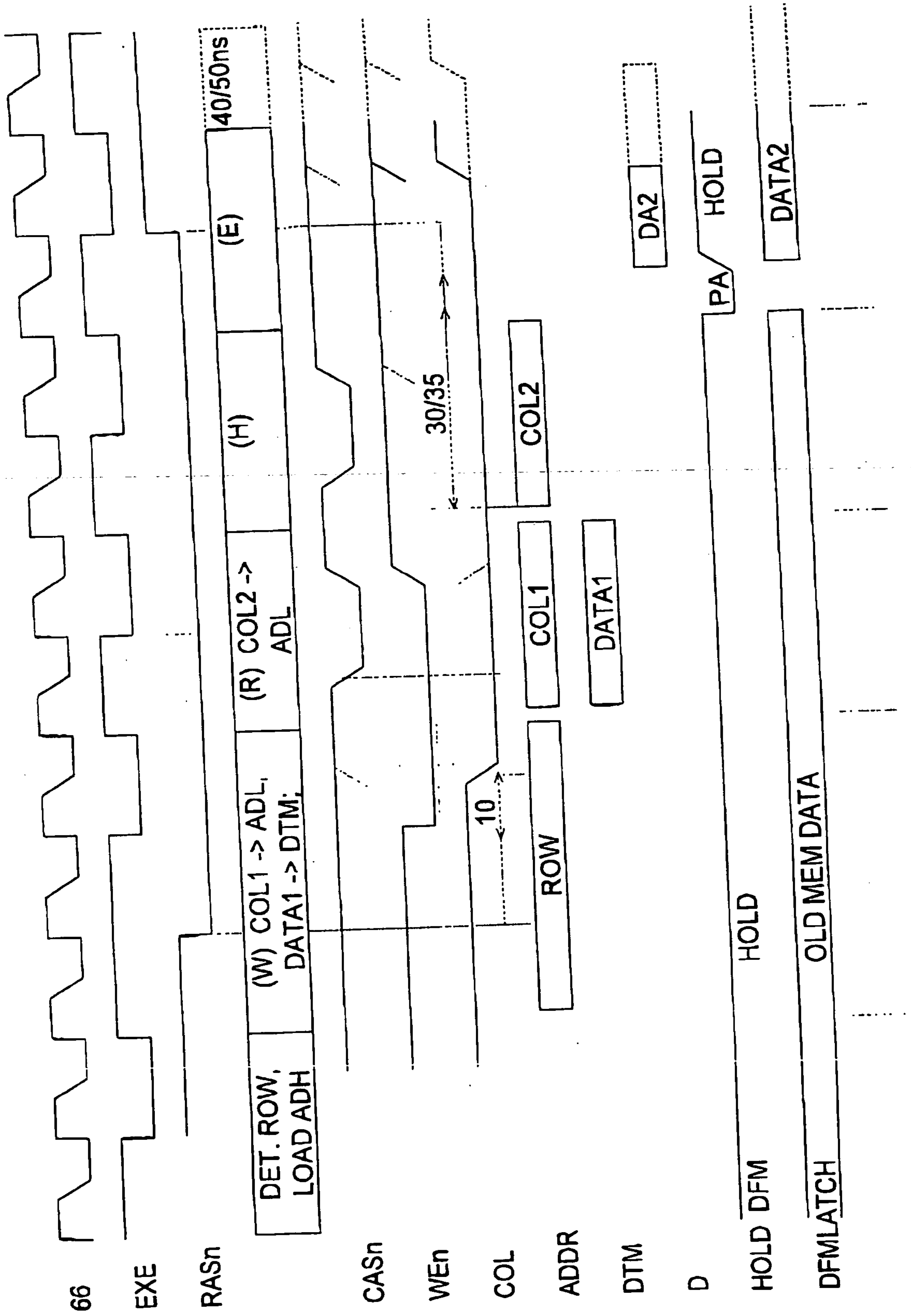
66
EXE
RASn
CASn
WEn
COL
ADDR
DTM
D
HOLD DFM
DFMLATCH
DET. ROW, LOAD ADH
(W) COL1 -> ADL, DATA1 -> DTM;
(R) COL2 -> ADL
(H)
(E)
40/50ns
30/35
10
ROW
COL1
COL2
DATA1
DA2
PA
HOLD
DATA2
HOLD
OLD MEM DATA

Fig. 7D

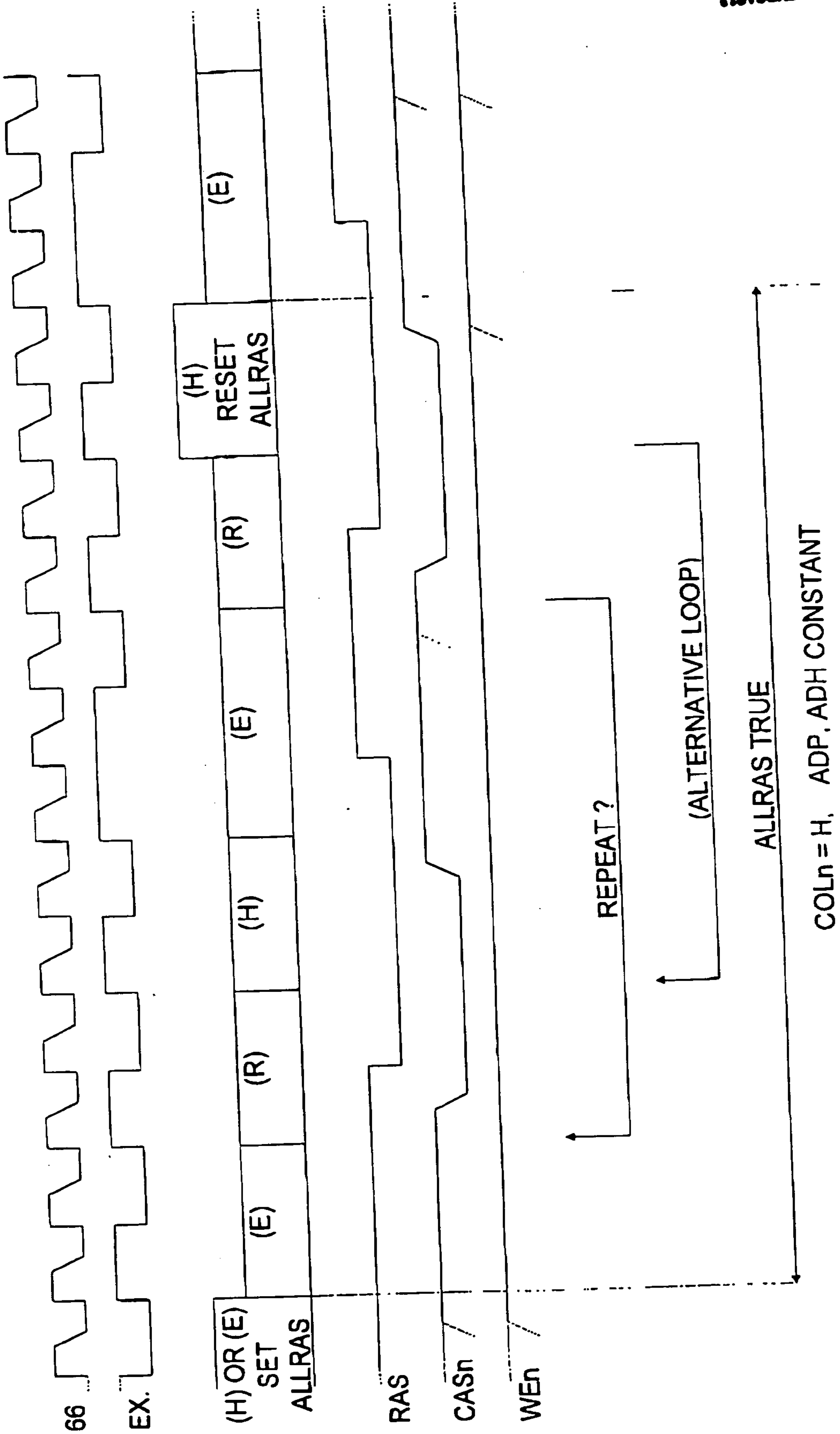
66
EX.
(H) OR (E) SET ALLRAS
RAS
CASn
WEn
(E)
(R)
(H)
(E)
(R)
(H) RESET ALLRAS
(E)
REPEAT ?
(ALTERNATIVE LOOP)
ALLRAS TRUE
COLn = H, ADP, ADH CONSTANT

Fig. 7E